Opérateur **EXEC_LOGICIEL**

1 But

Appeler depuis *Code_Aster*, un logiciel ou une commande système. Logiciel externe et commande système appelés peuvent résider et seront exécutés sur la même machine que *Code_Aster*, ou sur une machine distante. Il est plus particulièrement destiné au développement de macro-commandes. Cette macro python s'appuie sur le module python `os` et plus particulièrement sur la commande `os.system`.
Cet opérateur permet aussi d'engendrer directement depuis le fichier de commande un maillage de type GMSH, GIBI ou SALOME. Cette possibilité a été introduite essentiellement pour être utilisée dans les tests de non-régression du code, la vérification du maillage restant indispensable pour pouvoir lancer une étude.
Enfin, cette commande permet d'exécuter un script Python à la syntaxe SALOME dans une instance de SALOME, sur la machine courante ou sur une machine distante.

2 Syntaxe

```
EXEC_LOGICIEL (

    ◊  |  LOGICIEL  =  nom_exe,                                [TX]

       |  MAILLAGE  =  _F (
          ♦  FORMAT    =  /  'GMSH',                           [TX]
                          /  'GIBI',
                          /  'SALOME',
          ◊  UNITE_GEOM   =  igeom,                            [I]
          ◊  UNITE        =  unite,                            [I]
          ♦  MAILLAGE     =  ma,                               [maillage]
                          )

       |  MACHINE_DISTANTE =  _F (
          ♦  SSH_ADRESSE  =  'adresse',                        [TX]
          ◊  SSH_LOGIN    =  'login',                          [TX]
          ◊  SSH_PORT     =  'port',                           [TX]
                          )

       |  SALOME    =  _F (
          ♦  /  CHEMIN_SCRIPT    =  '/../script.py',           [TX]
             /  UNITE_SCRIPT     =  unite                      [I]

          ◊  SALOME_HOST         =  'adresse',                 [TX]
          ◊  SALOME_PORT         =  port,                      [I]
          ◊  SALOME_RUNAPPLI     =  '/../runSalomeScript',     [TX]
          ◊  FICHIERS_ENTREE     =  l_arg ,                    [l_TX]
          ◊  FICHIERS_SORTIE     =  l_arg ,                    [l_TX]
          ◊  /  NOM_PARA         =  l_arg ,                    [l_TX]
          ◊  /  VALE             =  l_arg ,                    [l_TX]
                          )

          ◊  ARGUMENT  =  l_arg,                               [l_TX]

          ◊  CODE_RETOUR_MAXI =  /  icode,                     [I]
                                 /  0,                         [DEFAUT]
          ◊  INFO   =  /  1  ,                                 [I]
                       /  2  ,                                 [DEFAUT]
          )
```

3 Opérandes

3.1 Opérande LOGICIEL

◊ LOGICIEL = nom_exe

Nom de la commande ou de l'exécutable à appeler. C'est une chaîne de caractères (entre '), il faut préciser l'ensemble du chemin (path) pour atteindre l'exécutable.
Cet opérande peut être utilisé pour surcharger la commande par défaut lors de la création d'un maillage.

Remarque :

Les logiciels référencés officiellement doivent posséder un point d'entrée sous un répertoire particulier géré par l'administrateur (répertoire outils de l'installation). Le développeur de macro-commande doit prendre la précaution de construire le nom en concaténant le nom du répertoire d'installation qu'il récupèrera à l'aide d'un utilitaire afin d'en assurer la portabilité.

3.2 Mot clé ARGUMENT

◊ ARGUMENT = l_arg

Permet de définir la liste des arguments passés à l'exécutable. Ce mot clé est obligatoire dans le cas où l'on crée un maillage au format SALOME/MED.
Les valeurs sont passées sous forme de chaînes de caractères, à charge de l'exécutable de les décoder, elles sont transmises telles quelles, le caractère ' ' (blanc) servant de séparateur.

3.3 Mot clé MAILLAGE

◊ MAILLAGE = _F (

Permet d'engendrer un maillage en appelant directement depuis le fichier de commandes l'un des outils suivants : GMSH, GIBI ou SALOME en fournissant le jeu de données dans un fichier. Les différents formats sont liés à la présence de la commande associée dans le répertoire d'installation de *Code_Aster*.

3.3.1 Opérande FORMAT

/ FORMAT = 'GMSH'

Création d'un maillage au format GMSH .

/ FORMAT = 'GIBI'

Création d'un maillage au format GIBI.

/ FORMAT = 'SALOME'

Création d'un maillage au format SALOME, ou plus généralement un fichier au format MED.
Dans ce cas le mot clé ARGUMENT est obligatoire et contient le nom du fichier MED produit par le script.

3.3.2 Opérande UNITE_GEOM

◊ UNITE_GEOM = igeom

Numéro d'unité logique associé au fichier de données utilisé pour créer le maillage.

Attention le fichier de données GIBI doit se terminer par la commande :

```
OPTI SAUV FORM 'fort.8'.
```

3.3.3 Opérande UNITE

◊ UNITE = unite

Numéro d'unité logique associé au fichier de résultat produit par l'outil de maillage.

3.3.4 Opérande MAILLAGE

♦ MAILLAGE = ma

Nom du concept maillage produit. Le nom doit être indiqué sous la forme syntaxique : CO ('MA')

3.4 Mot clé MACHINE_DISTANTE

◊ MACHINE_DISTANTE = _F (

Permet d'exécuter la commande définie par le mot-clé LOGICIEL sur une machine distante. Le protocole de communication réseau SSH sera utilisé pour la connexion à la machine distante, il est donc nécessaire que les comptes utilisateurs soient correctement configurés pour un accès sans mot de passe (utilisation de clé SSH).

3.4.1 Opérande SSH_ADRESSE

/ SSH_ADRESSE = 'adresse'

Il s'agit de l'adresse « réseau » de la machine : soit l'adresse IP (exemple 130.98.x.y sur le réseau interne EDF), soit le nom complet de la machine (exemple clau5aaa.der.edf.fr). Le nom court (ou *hostname*) peut également être suffisant si les deux machines sont sur le même réseau (exemple clau5aaa).

3.4.2 Opérande SSH_LOGIN

/ SSH_LOGIN = 'login'

Le login de l'utilisateur sur la machine distante. S'il n'est pas précisé, le même login que sur le serveur d'exécution d'Aster sera utilisé.

3.4.3 Opérande SSH_PORT

/ SSH_PORT = port

Ce mot-clé permet de redéfinir le port du serveur SSH. C'est une fonction avancée permettant de s'adapter à des environnements informatiques particuliers. Dans la majorité des cas, les utilisateurs n'ont pas à spécifier le port SSH, le port par défaut (22) étant utilisé.

3.5 Mot clé SALOME

◊ SALOME = _F (

Permet d'exécuter un script dans une instance de Salome, qui doit être lancée par ailleurs (Aster ne lance pas Salome), sur la même machine que la machine d'exécution d'Aster ou sur une machine distante. Le script Salome doit être à la syntaxe Python de Salome, c'est-à-dire que c'est un script qui peut s'exécuter depuis Salome via un *Load Script*. Il doit suivre un certain nombre de conventions d'écriture, notamment sur les variables utilisées pour les fichiers d'entrée (c'est-à-dire utilisés par le script) et les fichiers de sortie (c'est-à-dire générés par le script), car des ajustements/remplacements sont opérés avant l'exécution dans Salome.

3.5.1 Opérande CHEMIN_SCRIPT

◊ CHEMIN_SCRIPT = '/../script.py'

Ce mot-clé permet de spécifier le chemin du script Salome. On peut utiliser un chemin absolu (/home/user/mon-script.py) ou relatif (./fort.99 va ouvrir le fichier fort.99 contenu dans le répertoire temporaire d'exécution d'Aster).

3.5.2 Opérande UNITE_SCRIPT

◊ UNITE_SCRIPT = unite

Ce mot-clé permet de spécifier l'unité logique du script Salome. Cela permet d'intégrer le script Salome comme fichier d'entrée au profil d'exécution d'ASTK ou au fichier .export (utiliser le type libr associé à l'unité logique définie par ce mot-clé).

3.5.3 Opérande SALOME_HOST

◊ SALOME_HOST = 'adresse'

Ce mot-clé permet de spécifier l'adresse IP ou le nom de machine (suivant les mêmes règles que le mot-clé SSH_ADRESSE) sur laquelle est ouvert Salome. Si le mot-clé n'est pas spécifié, la machine locale sera utilisée.

3.5.4 Opérande SALOME_PORT

◊ SALOME_PORT = port

Ce mot-clé permet de spécifier le port de l'instance Salome à laquelle on cherche à se rattacher. Ce port est donné lors du lancement de Salome à partir d'un terminal :

```
assire@claut629:~$ /local00/salome/SALOME-MECA-2011.1/runSalomeMeca
Loading environnement for python 2.4
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME/SALOME5/V5_1_5/prerequis-V5_1_5.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME/SALOME5/V5_1_5/envSalome-V5_1_5.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME-MECA/V5_1_5/envSalomeMeca.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME-MECA/V5_1_5/prerequis-SalomeMeca.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME-MECA/V5_1_5/prerequis-Aster.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME-MECA/V5_1_5/prerequis-Eficas.sh
[OK] . /local00/salome/SALOME-MECA-2011.1/SALOME-MECA/V5_1_5/prerequis-OM.sh

Checking... LD_LIBRARY_PATH

Checking... PATH

Checking... PYTHONPATH
CHECK /home/assire
Searching for free port for the SALOME Naming Service: 2810 - Ok
runSalome running on claut629
```

La valeur par défaut est **2810**, mais si plusieurs instances de Salome tournent sur la machine, les autres utiliseront successivement les ports suivants : 2811, 2812, etc..

3.5.5 Opérande SALOME_RUNAPPLI

◊ SALOME_RUNAPPLI = '/../runSalomeScript'

Ce mot-clé permet de spécifier le lanceur de scripts dans l'environnement Salome. Ce lanceur fait partie de la distribution de Salome et est contenu dans le répertorie de Salome. S'il n'est pas spécifié, on utilisera le lien contenu dans le répertoire outils d'Aster, qui correspond à la version par défaut de Salome.

3.5.6 Opérande FICHIERS_ENTREE

◊ FICHIERS_ENTREE = ['/../fichier_in1', '/../fichier_in2', ...]

Ce mot-clé permet de spécifier la liste des fichiers de données du script Salome (par exemple un fichier MED si le script correspond à un post-traitement).
Pour que le script puisse fonctionner, et notamment à distance, il est nécessaire de suivre la convention d'écriture suivante : chaque fichier d'entrée doit apparaître dans le script Salome sous la forme de la variable : `INPUTFILE1=` (pour le premier fichier), `INPUTFILE2=` (pour le deuxième), etc.. :

```
INPUTFILE1 = toto
INPUTFILE2 = tutu
```

Les chaînes de caractères `toto` et `tutu` seront alors remplacées par les chemins des fichiers définis par `FICHIERS_ENTREE` avant l'exécution dans Salome.

3.5.7 Opérande FICHIERS_SORTIE

◊ `FICHIERS_SORTIE` = ['/../fichier_out1', '/../fichier_out2', ...]

Ce mot-clé permet de spécifier la liste des fichiers qui seront générés par le script Salome (par exemple un fichier MED si le script correspond à une opération de maillage).
La même convention que pour les fichiers d'entrée et de sortie doit être respectée pour le script : chaque fichier doit apparaître dans le script Salome sous la forme de la variable : `OUTPUTFILE1=` (pour le premier fichier), `OUTPUTFILE2=` (pour le deuxième), etc.. :

```
OUTPUTFILE1 = toto2
OUTPUTFILE2 = tutu2
```

Les chaînes de caractères `toto2` et `tutu2` seront alors remplacées par les chemins des fichiers définis par `FICHIERS_SORTIE` avant l'exécution dans Salome.

3.5.8 Opérande NOM_PARA

◊ `NOM_PARA` = ['para1', 'para2', ...]

Ce mot-clé permet de spécifier une liste de variables (liste de chaine de texte) dont les valeurs seront remplacées par les valeurs définies par le mot-clé `VALE`.
La même convention que pour les fichiers d'entrée et de sortie doit être respectée pour le script :

```
para1 = titi
para2 = tata
```

Les lignes contenant « `para1 =` » et « `para2 =` » (pour chaque paramètre, seule la première ligne rencontrée sera retenue) seront identifiées comme ligne à modifier à partir des données du mot-clé `VALE`.

3.5.9 Opérande VALE

◊ `VALE` = ['vale1', 'vale2', ...]

Ce mot-clé permet de spécifier les valeurs correspondant aux variables qui ont été spécifiées par `NOM_PARA`..
A partir de la convention d'écriture du script :

```
para1 = titi
para2 = tata
```

les blocs à droite du signe « égal » (`titi` et `tata`) seront remplacés par les valeurs définies dans la liste VALE :

```
para1 = vale1
para2 = vale2
```

3.6 Opérande INFO

◊ `INFO = info`

Dans le cas où `INFO=2`, les messages provenant de la commande exécutée sont imprimés dans le fichier `MESSAGE`. C'est la valeur par défaut. Cela permet de conserver la trace de l'exécution lors de l'utilisation `EXEC_LOGICIEL/MAILLAGE`, particulièrement dans les cas-tests.

3.7 Opérande CODE_RETOUR_MAXI

◊ `CODE_RETOUR_MAXI = icode`

Valeur maximum du code retour renvoyé par la commande ou le logiciel qui est tolérée pour considérer que l'exécution s'est bien déroulée. Par défaut cette valeur vaut 0, si elle est affectée à –1, le code retour de la commande ou du logiciel est ignoré.

4 Exemples

`EXEC_LOGICIEL` n'est pas la seule solution pour appeler une commande ou un programme, il est possible en insérant une commande python `os.system` d'effectuer le même type d'opération.

4.1.1 Exemple de lancement d'une commande

Appel d'une commande unix de base :

```
EXEC_LOGICIEL(LOGICIEL='ls -la', ARGUMENT='/tmp',)
```

```
Lancement à distance :
```

```
EXEC_LOGICIEL(LOGICIEL='ls -al', ARGUMENT='/tmp',
              MACHINE_DISTANTE=_F(SSH_ADRESSE  = 'clau5aaa.der.edf.fr',
                                  SSH_PORT     = 22,),);
```

4.1.2 Exemple de création d'un maillage

Création d'un maillage :

```
EXEC_LOGICIEL(MAILLAGE=_F(FORMAT='GIBI',
                          UNITE_GEOM=17, UNITE=18,
                          MAILLAGE=CO('magibi'),),
             LOGICIEL='gibi'
             CODE_RETOUR_MAXI = 2,
             INFO=2,)
```

Création d'un maillage au format SALOME.

```
EXEC_LOGICIEL(MAILLAGE=_F(FORMAT='SALOME',
                          UNITE_GEOM=15,
                          UNITE=21,
                          MAILLAGE=CO('mamed'),),
             ARGUMENT='cube.mmed',
             INFO=1)
```

4.1.3 Exemple de lancement d'un script Salome

La portion de script suivante est donnée en exemple :

```
# Script /home/assire/test.py (post-traitement dans Salome)
import salome
import VISU

INPUTFILE1  = '/home/assire/toto'
OUTPUTFILE1 = ''
CHOIX       = '$CHOIX$'
PARA2       =

(suite du script)
```

La suite de commande suivante :

```
IMPR_RESU(FORMAT='MED',
          UNITE=90,
          RESU=_F(RESULTAT=res,),);

EXEC_LOGICIEL(
   SALOME=_F(CHEMIN_SCRIPT   = '/home/assire/test.py',
             SALOME_HOST     = 'cli75ca.der.edf.fr',
             SALOME_PORT     = 2811,
             FICHIERS_ENTREE = [ './fort.90' ],
             FICHIERS_SORTIE = [ './fort.98' ],
             SALOME_RUNAPPLI = '/chemin-salome/runSalomeScript',
             NOM_PARA = [ 'CHOIX', 'PARA2' ],
             VALE     = [ 'DEPL', '10' ],),
   INFO=2);
```

Permet d'envoyer le script de post-traitement et le fichier MED sur la machine distante et lancer le script dans Salome, avant de récupérer le résultat du script dans le fichier fort.98.

Le script modifié par `EXEC_LOGICIEL` et exécuté dans Salome aura la forme :

```
# Script /home/assire/test.py (post-traitement dans Salome)
import salome
import VISU

INPUTFILE1  = '/tmp/fort.90'
OUTPUTFILE1 = '/tmp/fort.98'
CHOIX       = 'DEPL'
PARA2       = 10

(suite du script)
```